PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

| | |
|---|---|
| In re Application of: Padmanabhan Sreenivasan et al. | Examiner: Joseph E. Avellino |
| Serial No.: 09/997,404 | Group Art Unit: 2143 |
| Filed: November 29, 2001 | Docket: 499.074US2 |

For: FLEXIBLE FAILOVER POLICIES IN HIGH AVAILABILITY COMPUTING SYSTEMS

---

## APPEAL BRIEF UNDER 37 CFR § 41.37

Mail Stop Appeal Brief- Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

The Appeal Brief is presented in support of the Notice of Appeal to the Board of Patent Appeals and Interferences, filed on August 3, 2007, from the Final Rejection of claims 1, 2, 4, 5, 7-12 and 14-19 of the above-identified application, as set forth in the Final Office Action mailed on May 3, 2007.

The Commissioner of Patents and Trademarks is hereby authorized to charge Deposit Account No. 19-0743 in the amount of $500.00 which represents the requisite fee set forth in 37 C.F.R. § 41.20(b)(2). The Appellants respectfully request consideration and reversal of the Examiner's rejections of pending claims.

# APPEAL BRIEF UNDER 37 C.F.R. § 41.37

## TABLE OF CONTENTS

Page

1. REAL PARTY IN INTEREST ..... 3
2. RELATED APPEALS AND INTERFERENCES ..... 4
3. STATUS OF THE CLAIMS ..... 5
4. STATUS OF AMENDMENTS ..... 6
5. SUMMARY OF CLAIMED SUBJECT MATTER ..... 7
6. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL ..... 9
7. ARGUMENT ..... 10
8. CLAIMS APPENDIX ..... 17
9. EVIDENCE APPENDIX ..... 20
10. RELATED PROCEEDINGS APPENDIX ..... 21

## 1. REAL PARTY IN INTEREST

The real party in interest of the above-captioned patent application is the assignee, SILICON GRAPHICS, INC.

## 2. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellant that will have a bearing on the Board's decision in the present appeal.

## 3. STATUS OF THE CLAIMS

The present application was filed on November 29, 2001 with claims 1-2 and claimed priority to Unites States Patent Application Serial No 09/811,357, filed March 16, 2001, now abandoned.

A non-final Office Action was mailed January 4, 2005, rejecting claims 1 and 2. A response to the non-final Office Action was filed July 5, 2005, amending claims 1 and 2, and adding new claims 3-13.

A non-final Office Action was mailed November 1, 2005 rejecting claims 1-13. A response to the non-final Office Action was filed April 3, 2006, amending claims 1 and 2, and adding claims 14-18.

A final Office Action was mailed May 3, 2006, rejecting claims 1-18. A Request for Continued Examination was filed September 28, 2006 along with an amendment amending claims 1-2, 4, 7-10 and 18, cancelling claims 3, 6 and 13, and adding new claim 19.

A non-final Office Action was mailed November 28, 2006 rejecting claims 1-19 (note the cancellation of claims 3, 6 and 13 was not addressed). A response to the non-final Office Action was filed March 28, 2007, amending claims 15 and 17.

A final Office Action (hereinafter "the Final Office Action") was mailed May 3, 2007, rejecting claims 1-2, 4-5, 7-12 and 14-19. Claims 1-2, 4-5, 7-12 and 14-19 stand twice rejected, remain pending, and are the subject of the present Appeal.

## 4. STATUS OF AMENDMENTS

No amendments have been made subsequent to the Final Office Action dated May 3, 2007.

## 5. SUMMARY OF CLAIMED SUBJECT MATTER

A first aspect of the claimed subject matter includes a system for implementing a failover policy. The system may include a cluster infrastructure for managing a plurality of nodes, a high availability infrastructure for providing group and cluster membership services, and a high availability script execution component operative to receive a failover script and at least one failover attribute and operative to produce a failover domain.

A further aspect of the claimed subject matter includes a method for determining a target node for a failover comprises by executing a failover script that produces a failover domain, the failover domain having an ordered list of nodes, receiving a failover attribute and based on the failover attribute and failover domain, selecting a node upon which to locate a resource.

Further details on the claimed subject matter are presented below.

### INDEPENDENT CLAIM 1

1. A system for implementing a failover policy comprising:

a cluster infrastructure for managing a plurality of nodes; *[FIG. 1B, element 12; page 11, lines 3-12]*

a high availability infrastructure for providing group and cluster membership services; *[FIG. 1B, element 14; page 11, lines 13-34]* and

a high availability script execution component *[FIG. 1B, element 16; page 11, line 35 – page 12, line5; page 14, lines 3-6]* operative upon the detection of a failover event to perform the tasks of:

receive a failover script comprising a set of one or more commands and further operable to receive at least one failover attribute and operative to cause the failover script to be interpreted to produce a run-time failover domain from an initial failover domain, *[FIG. 2, elements 202-206; page 14, line 3 – page 15, line 14]* and

execute one or more action scripts, the action scripts when executed causing a resource group to failover to a node in the run-time failover domain, the resource group having one or more resources. *[page 9, line 25 – page 10, line 12]*

## INDEPENDENT CLAIM 2

2. A method comprising:

detecting a failover event; *[page 14, lines 3-5]*

upon detecting the failover event, executing a failover script, said script comprising a set of one or more commands that when executed determine a run-time failover domain from an initial failover domain, said run-time failover domain having an ordered list of nodes; *[FIG. 2, elements 202-204; page 14, line 3- page 15, line 8]*

receiving a failover attribute; *[FIG. 2, element 206; page 15, lines 9-11]*

based on the failover attribute and run-time failover domain, selecting a node upon which to locate a resource; *[FIG. 2, element 208; page 15, lines 11-14]* and

executing one or more action scripts, the action scripts causing a resource group to failover to the selected node. *[page 9, line 25 – page 10, line 12]*

## DEPENDENT CLAIM 7

7. The method of claim 2, wherein the action script verifies that the resource is configured on the target node. *[page 14, line 11]*

## DEPENDENT CLAIM 19

19. The system of claim 1, wherein the one or more resources includes an application and further comprising an application plug-in that provides a high-availability interface for the application. *[FIG. 1B, element 18; page 12, line 16 – page 13, line5]*

This summary does not provide an exhaustive or exclusive view of the present subject matter, and Appellant refers to each of the appended claims and its legal equivalents for a complete statement of the invention.

## 6. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1, 2, 4, 5, 7-12 and 14-19 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Alexander et al. (U.S. Patent No. 6,189,111) in view of Le et al. (U.S. Patent No. 6,145,089) in view of Chao et al. (U.S. Patent No. 6,438,705).

## 7. ARGUMENT

### A. The Rejections under 35 U.S.C. § 103

#### 1) The Applicable Law under 35 U.S.C. § 103

The determination of obviousness under 35 U.S.C. § 103 is a legal conclusion based on factual evidence. *See Princeton Biochemicals, Inc. v. Beckman Coulter, Inc.*, 411 F.3d 1332, 1336-37 (Fed.Cir. 2005). The legal conclusion that a claim is obvious within § 103(a) depends on at least four underlying factual issues set forth in *Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 17, 86 S.Ct. 684, 15 L.Ed.2d 545 (1966). The underlying factual issues set forth in *Graham* are as follows: (1) the scope and content of the prior art; (2) differences between the prior art and the claims at issue; (3) the level of ordinary skill in the pertinent art; and (4) evaluation of any relevant secondary considerations.

The Examiner has the burden under 35 U.S.C. § 103 to establish a *prima facie* case of obviousness. *In re Fine*, 837 F.2d 1071, 1074, 5 USPQ2d 1596, 1598 (Fed. Cir.1988). To establish *prima facie* obviousness of a claimed invention, all the claim limitations must be taught or suggested, by the prior art. *In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974) ; M.P.E.P. § 2143.03. "All words in a claim must be considered in judging the patentability of that claim against the prior art." *In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970) ; M.P.E.P. § 2143.03. As part of establishing a *prima facie* case of obviousness, the Examiner's analysis must show that some objective teaching in the prior art or that knowledge generally available to one of ordinary skill in the art would lead an individual to combine the relevant teaching of the references. *Id.* To facilitate review, this analysis should be made explicit. *KSR Int'l v. Teleflex Inc., et al.*, 127 S.Ct. 1727; 167 L.Ed 2d 705; 82 USPQ2d 1385 (2007) (citing *In re Kahn*, 441 F. 3d 977, 988 (Fed. Cir. 2006)).

The Federal Circuit has stated:

> Obviousness is tested by "what the combined teaching of the references would have suggested to those of ordinary skill in the art." *In re Keller*, 642 F.2d 413, 425, 208 USPQ 871, 878 (CCPA 1981)). But it "cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination." *ACS Hosp. Sys.*, 732 F.2d at 1577, 221 USPQ at 933. And

> "teachings of references can be combined *only* if there is some suggestion or incentive to do so." *Id.* (emphasis in original).
>
> *In re Fine*, 837 F.2d 1071; 5 USPQ2d 1596 (Fed. Cir.1988).

The test for obviousness under §103 must take into consideration the invention as a whole; that is, one must consider the particular problem solved by the combination of elements that define the invention. *Interconnect Planning Corp. v. Feil*, 774 F.2d 1132, 1143, 227 USPQ 543, 551 (Fed. Cir.1985). The Examiner must, as one of the inquiries pertinent to any obviousness inquiry under 35 U.S.C. §103, recognize and consider not only the similarities but also the critical differences between the claimed invention and the prior art. *In re Bond*, 910 F.2d 831, 834, 15 USPQ2d 1566, 1568 (Fed. Cir. 1990), *reh'g denied*, 1990 U.S. App. LEXIS 19971 (Fed. Cir.1990). The fact that a reference teaches away from a claimed invention is highly probative that the reference would not have rendered the claimed invention obvious to one of ordinary skill in the art. *Stranco Inc. v. Atlantes Chemical Systems, Inc.*, 15 USPQ2d 1704, 1713 (Tex. 1990). When the prior art teaches away from combining certain known elements, discovery of a successful means of combining them is more likely to be nonobvious. *KSR Int'l v. Teleflex Inc., et al.*, 127 S.Ct. 1727; 167 L.Ed 2d 705; 82 USPQ2d 1385 (2007).

Further, conclusions of obviousness must be based on facts, not generality. *In re Warner*, 379 F.2d 1011, 1017 (C.C.P.A. 1967); *In re Freed*, 425 F.2d 785, 787 (C.C.P.A. 1970). In fact, there must be a rational underpinning grounded in evidence to support the legal conclusion of obviousness. The Federal Circuit has stated that, "rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." *In re Kahn,* 441 F. 3d 977, 988 (Fed. Cir. 2006), citing *In re Lee*, 61 USPQ2d 1430 (Fed. Cir.2002); 72 FR 57527-28 (Oct. 10, 2007).

Moreover, "mere identification in the prior art of each element is insufficient to defeat the patentability of the combined subject matter as a whole." *In re Kahn,* 441 F. 3d 977, 988 (Fed. Cir. 2006). This was recently echoed by the U.S. Supreme Court in *KSR Int'l v. Teleflex Inc., et al.*, 127 S.Ct. 1727; 167 L.Ed 2d 705; 82 USPQ2d 1385 (2007) (a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art.).

## 2) The Application of 35 U.S.C. § 103(a) to the Rejected Claims

### A) The Rejection of claims 1 and 2 under 35 U.S.C. § 103(a).

Claims 1 and 2 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Alexander et al. (U.S. Patent No. 6,189,111; hereinafter "Alexander") in view of Le et al. (U.S. Patent No. 6,145,089; hereinafter "Le") in view of Chao et al. (U.S. Patent No. 6,438,705; hereinafter "Chao"). Appellant respectfully traverses the rejection because of the differences between claims 1 and 2 and the cited references. In particular, claims 1 and 2 recite elements not found in the combination of Alexander, Le and Chao.

For example, claim 1 recites a high availability script execution component that is operable "receive a failover script comprising a set of one or more commands and further operable to receive at least one failover attribute and operative to cause the failover script to be interpreted to produce a run-time failover domain from an initial failover domain." Claim 2 recites similar language. The Final Office Action, with respect to the recited language, first states that Alexander at column 6, lines 40-50 teaches "a high availability script execution component (i.e. routine execution mechanism)." Appellant respectfully disagrees with this interpretation of Alexander. The cited section of Alexander merely makes a generalized comment that a routine may be executed to investigate an error and take an action. Nowhere in the cited section nor in Alexander as a whole is a script execution component taught that interprets a failover script. In fact, the phrase "execution of a routine" appears to teach away from the use of scripts in view of the execution of binary code (e.g. a subroutine) rather than script execution.

Further, the Final Office Action states that Alexander, at column 5 lines 40-50, column 6 lines 19-21, column 8 lines 40-42 and column 9 lines 32-33 teaches producing a runtime failover domain from an initial failover domain because Alexander teaches "recognizing the failing node and removing it from the bitmap." Appellant notes that nowhere does Alexander teach that the bitmap is manipulated by a failover script that is interpreted to produce a run-time failover domain. Rather, the Alexander teaches the rote removal of a failing node. Alexander does not teach that a script is interpreted and that the script provides logic to intelligently produce a

failover domain. The intelligent selection of a failover domain through the use of a failover script provides advantages over Alexander in that a failover domain may be produced that includes machines that are best able to take over for the failed node rather than the rote inclusion of any remaining node in the cluster as reflected by a bitmap.

The Final Office Action correctly states that Alexander does not "specifically state that upon the detection of a failover event, executing a failover script comprising a set of one or more commands." However, the Office Action attempts to make up for the deficiency by stating that Le, at column 4, lines 37-60, column 5, lines 53-67, and FIG 4 ref. 450 teaches a failover script. Appellant respectfully disagrees with this interpretation of Le. The failover script as defined and properly interpreted in the claims includes commands that provide a user the opportunity to define commands that will determine a run-time failover domain, thereby providing for flexibility in determining a run-time failover domain. None of the scripts mentioned in Le produce a run-time failover domain. Instead, the scripts in Le are more akin to action scripts in that they "start, stop, and restart a service or services" (see column 4, lines 37-52). Thus Le fails to teach a failover script.

In the "Response to Arguments" section, the Final Office Action states that the scripts in Le are failover scripts, and states that Le discloses that the script "includes a response to service failure and may include scripts to start, stop and restart the device." (emphasis added). Le does not disclose any scripts that start, stop or restart a device, rather the correct language from Le is that the scripts "may contain scripts to start, stop, and restart a service or services within the service group." (emphasis added). Here, the service group can be considered a domain. However, nowhere does Le teach or suggest that scripts are used to determine which nodes are in the domain. Rather the scripts control starting stopping and restarting services, exactly the types of actions defined as "action scripts" in Appellant's specification. Further, the mere statement in Le that a script is executed in response to a service failure does not teach or suggest that the scripts determine a failover domain as recited in claims 1 and 2.

Appellant has reviewed Chao and can find no teaching or suggestion of a failover script that receives an input domain and is interpreted to produce a run-time failover domain.

In view of the above, the combination Alexander, Le and Chao fails to teach each and every element of claims 1 and 2. Thus claims 1 and 2 are not obvious in view of the

combination of Alexander, Le and Chao. Appellant respectfully requests reversal of the rejection of claims 1 and 2.

**B) The Rejection of claims** 7-12 and 14-19 **under 35 U.S.C. § 103(a).**

Claim 7 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Alexander in view of Le and in view of Chao. Appellant respectfully traverses the rejection because of the differences between claim 7 and the cited references. In particular, claim 7 recites elements not found in the combination of Alexander, Le and Chao.

For example, claim 7 recites that "the action script verifies that the resource is configured on the target node." The Final Office Action states that Alexander, at column 9 lines 18-25 teaches the recited language in stating that Alexander's "ASSERT macros that perform consistency checks and can be used on candidates for harvesting; and can further be used by applications for validation purposes." Appellant respectfully disagrees. As clearly stated in the cited section, an ASSERT macro performs ad-hoc consistency checks and causes a panic (i.e. a shutdown) if the asserted condition is not true. Kernel or application code that checks for internal consistency of data structures is entirely different from executing an action script that verifies a resource is configured or not already running on an external target node. As a result, Alexander does not teach or suggest the recited language.

In the Response to Arguments section, the Final Office Action argues that the fact that an application can validate its own data structures includes verifying configuration on other devices. The Final Office Action fails to describe how it is either inherent or necessary that validating a data structure implies verifying that a resource is configured on a target node. Appellant respectfully submits that validating an application data structure using an "ASSERT" macro, whether done by an application or a kernel, does not teach or suggest anything about verifying whether a resource is configured on a target node.

For the reasons above, Appellant respectfully requests reversal of the rejection of claim 7.

**C) The Rejection of claims 19 under 35 U.S.C. § 103(a).**

Claim 19 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Alexander in view of Le and in view of Chao. Appellant respectfully traverses the rejection because of the differences between claim 19 and the cited references. In particular, claim 19 recites elements not found in the combination of Alexander, Le and Chao.

For example, claim 19 recites "an application plug-in that provides a high-availability interface for the application." The Office Action asserts that Alexander, at column 3, lines 8-20 discloses the recited language, interpreting a plug-in as "an interface utilized to communicate with a user or client application." Appellant respectfully disagrees with this interpretation of Alexander. First, nowhere does Alexander teach or suggest an application plug-in. The section of Alexander cited in the Office Action does not refer to interfaces, and nowhere does the term "plug-in" appear in Alexander. In fact, the cited section of Alexander explicitly states that the operating system "handles the details of processing a system failure when detected." Thus Alexander in fact teaches away from the use of application plug-ins to provide a high-availability interface to convert an application to a high availability application. In view of the above, none of Alexander, Le or Chao teach or suggest the use of an application plug-in to convert an application into a high-availability application. Thus the combination fails to teach or suggest each and every element of claim 19. Appellant respectfully requests reversal of the rejection of claim 19.

## SUMMARY

It is respectfully submitted that for the above reasons, the cited references do not render the claims obvious and that the claims are patentable over the cited art. Reversal of the rejections and allowance of the pending claims are respectfully requested.

Respectfully submitted,

PADMANABHAN SREENIVASAN et al.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. Box 2938
Minneapolis, MN 55402

Date March 3, 2008 By

Rodney L. Lacy
Reg. No. 41,136

**CERTIFICATE UNDER 37** CFR 1.8: The undersigned hereby certifies that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to: Mail Stop Appeal - Patent, Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on this 3rd day of March 2008.

Rodney L. Lacy
Name

Signature

## 8. CLAIMS APPENDIX

1. A system for implementing a failover policy comprising:

a cluster infrastructure for managing a plurality of nodes;

a high availability infrastructure for providing group and cluster membership services; and

a high availability script execution component operative upon the detection of a failover event to perform the tasks of:

receive a failover script comprising a set of one or more commands and further operable to receive at least one failover attribute and operative to cause the failover script to be interpreted to produce a run-time failover domain from an initial failover domain, and

execute one or more action scripts, the action scripts when executed causing a resource group to failover to a node in the run-time failover domain, the resource group having one or more resources.

2. A method comprising:

detecting a failover event;

upon detecting the failover event, executing a failover script, said script comprising a set of one or more commands that when executed determine a run-time failover domain from an initial failover domain, said run-time failover domain having an ordered list of nodes;

receiving a failover attribute;

based on the failover attribute and run-time failover domain, selecting a node upon which to locate a resource; and

executing one or more action scripts, the action scripts causing a resource group to failover to the selected node.

4. The method of claim 2, further comprising:

defining a set of resources for inclusion in the resource group; and

associating the failover script and the failover attribute with the resource group.

5. The method of claim 2, wherein selecting a node comprises selecting a first node in the ordered list of nodes.

7. The method of claim 2, wherein the action script verifies that the resource is configured on the target node.

8. The method of claim 2, wherein the action script verifies that the resource is not already running on the target node.

9. The method of claim 2, wherein the action script starts the resource.

10. The method of claim 2, wherein the action script stops the resource.

11. The system of claim 1, wherein the script is a shell script.

12. The system of claim 1, wherein the script is a Perl script.

14. The system of claim 1, wherein the failover event comprises failure of a node.

15. The system of claim 1, wherein the failover event comprises a load-balancing event independent of node failure.

16. The method of claim 2, wherein the failover event comprises failure of a node.

17. The method of claim 2, wherein the failover event comprises a load balancing event independent of node failure.

18. The method of claim 1, further comprising:

saving the run-time failover domain;

detecting a second failover event; and

executing the failover script upon detection of the second failover event, wherein the run-time failover domain is provided as input to the failover script and further wherein the failover script determines a second run-time failover domain.

19. The system of claim 1, wherein the one or more resources includes an application and further comprising an application plug-in that provides a high-availability interface for the application.

## 9. EVIDENCE APPENDIX

None.

## 10. RELATED PROCEEDINGS APPENDIX

None.